



## INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification <sup>7</sup> :

G06F 9/38

A1

(11) International Publication Number:

WO 00/39669

(43) International Publication Date:

6 July 2000 (06.07.00)

(21) International Application Number: PCT/US99/30480

(22) International Filing Date: 20 December 1999 (20.12.99)

(30) Priority Data:

09/220,154

23 December 1998 (23.12.98) US

(71) Applicant: INTEL CORPORATION [US/US]; 2625 Walsh Avenue, Santa Clara, CA 95051 (US).

(72) Inventors: CORWIN, Michael, P.; 1530 Nuthatch Lane, Sunnyvale, CA 94087 (US). SHARANGPANI, Harshvardhan; 558 Hubbard Avenue, Santa Clara, CA 95051 (US). MULLER, Hans; 199 Montcalm Street, San Francisco, CA 94110 (US). ARORA, Ken; 17445 Skyline Boulevard, Los Gatos, CA 95033 (US).

(74) Agent: VIKSNINS, Ann, S.; Schwegman, Lundberg, Woessner &amp; Kluth, P.O. Box 2938, Minneapolis, MN 55402 (US).

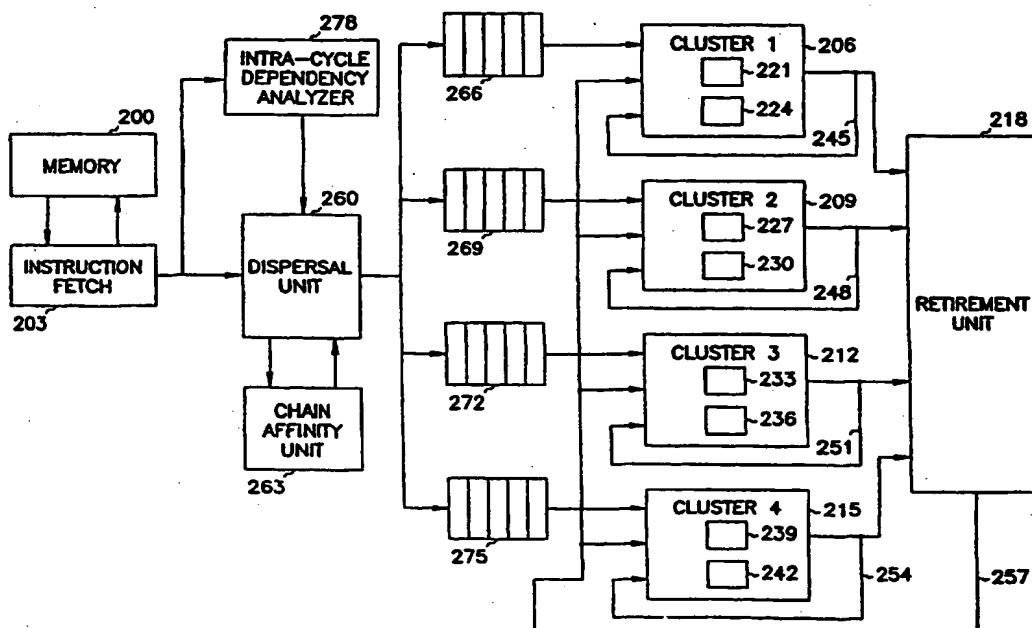
(81) Designated States: AE, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CR, CU, CZ, DE, DK, DM, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, TZ, UA, UG, UZ, VN, YU, ZA, ZW, ARIPO patent (GH, GM, KE, LS, MW, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

## Published

With international search report.

Before the expiration of the time limit for amending the claims and to be republished in the event of the receipt of amendments.

(54) Title: ROUTING DEPENDENT INSTRUCTIONS TO CLUSTERED EXECUTION UNITS



(57) Abstract

A dispersal unit in combination with a chain affinity unit and an intra-cycle dependency analyzer routes instructions in a microprocessor in order to improve microprocessor performance. The dispersal unit routes instructions to a particular cluster in the microprocessor in response to information stored in the chain affinity unit. The intra-cycle dependency analyzer identifies dependencies in groups of instructions to the dispersal unit, and the dispersal unit routes instructions in the group based on those dependencies.

**FOR THE PURPOSES OF INFORMATION ONLY**

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania	ES	Spain	LS	Lesotho	SI	Slovenia
AM	Armenia	FI	Finland	LT	Lithuania	SK	Slovakia
AT	Austria	FR	France	LU	Luxembourg	SN	Senegal
AU	Australia	GA	Gabon	LV	Latvia	SZ	Swaziland
AZ	Azerbaijan	GB	United Kingdom	MC	Monaco	TD	Chad
BA	Bosnia and Herzegovina	GE	Georgia	MD	Republic of Moldova	TG	Togo
BB	Barbados	GH	Ghana	MG	Madagascar	TJ	Tajikistan
BE	Belgium	GN	Guinea	MK	The former Yugoslav Republic of Macedonia	TM	Turkmenistan
BF	Burkina Faso	GR	Greece			TR	Turkey
BG	Bulgaria	HU	Hungary	ML	Mali	TT	Trinidad and Tobago
BJ	Benin	IE	Ireland	MN	Mongolia	UA	Ukraine
BR	Brazil	IL	Israel	MR	Mauritania	UG	Uganda
BY	Belarus	IS	Iceland	MW	Malawi	US	United States of America
CA	Canada	IT	Italy	MX	Mexico	UZ	Uzbekistan
CF	Central African Republic	JP	Japan	NE	Niger	VN	Viet Nam
CG	Congo	KE	Kenya	NL	Netherlands	YU	Yugoslavia
CH	Switzerland	KG	Kyrgyzstan	NO	Norway	ZW	Zimbabwe
CI	Côte d'Ivoire	KP	Democratic People's Republic of Korea	NZ	New Zealand		
CM	Cameroon			PL	Poland		
CN	China	KR	Republic of Korea	PT	Portugal		
CU	Cuba	KZ	Kazakstan	RO	Romania		
CZ	Czech Republic	LC	Saint Lucia	RU	Russian Federation		
DE	Germany	LI	Liechtenstein	SD	Sudan		
DK	Denmark	LK	Sri Lanka	SE	Sweden		
EE	Estonia	LR	Liberia	SG	Singapore		

## **Routing Dependent Instructions to Clustered Execution Units**

5

### Technical Field of the Invention

This invention relates to the field of microprocessor architecture, more particularly to an architecture that makes efficient use of instruction execution units in a multi-cluster system.

### Background of the Invention

10

Early microprocessors operated at relatively low clock frequencies. As users demanded faster microprocessors, designers responded by increasing the clock frequency. In some designs, the higher clock frequency did not interfere with the correct logical operation of the microprocessor. In other designs, the higher clock frequency caused subsystems in the microprocessor to fail. These failures were addressed in several ways. Some failures were corrected by packing the logic devices more densely on the chip in order to decrease signal path lengths between the logic devices. Others were corrected by implementing the design in a faster technology, such as gallium arsenide. As clock frequencies continued to increase, these strategies became more difficult and costly to implement, and other strategies evolved to satisfy the user's demand for faster microprocessors.

15

20

One such strategy involved designing multiple instruction execution units into a single microprocessor. A microprocessor having multiple instruction execution units can execute more instructions per unit of time than a microprocessor having a single instruction execution unit. This strategy evolved to a point where multiple instruction execution units were grouped or clustered to further increase microprocessor performance. However, the performance improvement in these multi-cluster microprocessors comes at the cost of increased complexity in the scheduler, the microprocessor subsystem that routes instructions to the clusters in an attempt to improve the utilization of the instruction execution units. An additional problem arises when the results produced by a first cluster are required for use by a second cluster. In that case, a delay in waiting for the results produced by the first cluster to be available to the second cluster reduces the throughput of the microprocessor.

25

30

Referring to FIG. 1, a block diagram of a prior art microprocessor system is shown. Memory 100 is provided for storing instructions. Coupled to memory 100 is instruction fetch 105. The purpose of instruction fetch 105 is to retrieve instructions from memory 100 and present them to scheduler 110. Scheduler 110 routes instructions to either first cluster 115 or second cluster 120. First execution unit 125 and second execution unit 130 are provided for executing instructions routed to first cluster 115. Third execution unit 135 and fourth execution unit 140 are provided for executing instructions routed to second cluster 120. Retirement unit 145 is coupled to the outputs of first cluster 115 and second cluster 120 and couples the architectural state via write back bus 160 to first cluster 115 and second cluster 120. The architectural state is the bit configuration of all the registers in retirement unit 145 at a given time. First cluster fast results bypass 150 is provided to couple the output of first cluster 115 to the input of first cluster 115, for use in first cluster 115, prior to commitment in retirement unit 145. Likewise, second cluster fast results bypass 155 is provided to couple the output of second cluster 120 to the input of second cluster 120, for use in second cluster 120, prior to commitment in retirement unit 145.

In operation, instruction fetch 105 retrieves instructions from memory 100 and delivers the instructions to scheduler 110. Scheduler 110 attempts to route instructions to first cluster 115 and second cluster 120 in a way that provides high utilization of execution units 125, 130, 135, and 140. Unfortunately, when a read instruction is executed in second cluster 120 after a write instruction was executed in first cluster 115, the results of the write instruction are not immediately available to the read instruction, since the results of the write instruction must be fed back to second cluster 120 from the architectural state in retirement unit 145 via write back bus 160.

For these and other reasons there is a need for the present invention.

#### Summary of the Invention

In one embodiment an apparatus for routing computer instructions comprises a plurality of queues to buffer instructions to a plurality of clusters, a chain affinity unit to store information, and a dispersal unit to route instructions to the plurality of queues based on information to be stored in the chain affinity unit.

### Brief Description of the Drawings

FIG. 1 is a block diagram of a prior art general purpose microprocessor system.

FIG. 2 is a block diagram of one embodiment of a microprocessor system  
5 of the present invention.

### Detailed Description of the Invention

Referring to FIG. 2, a block diagram of one embodiment of a microprocessor system of the present invention is shown. Memory 200 stores instructions, and instruction fetch 203 retrieves instructions from memory 200.  
10 Intra-cycle dependency analyzer 278 analyzes groups of instructions received from instruction fetch 203 and identifies dependent instructions in the analyzed group to dispersal unit 260. Dispersal unit 260 receives instructions from instruction fetch 203, manages chain affinity unit 263, and routes instructions received from instruction fetch 203 to a plurality of queues 266, 269, 272, and  
15 275. The plurality of queues is coupled to a plurality of clusters 206, 209, 212, and 215, which include eight execution units 221, 224, 227, 230, 233, 236, 239, and 242, and four fast results bypasses 245, 248, 251, and 254. The output of the clusters are coupled to retirement unit 218, which feeds back the architectural state via write back bus 257 to the clusters. Those skilled in the art will  
20 recognize that this embodiment of the present invention is not limited in the number of clusters, the number of execution units per cluster, or in the number of queues.

Dispersal unit 260, in one embodiment of the present invention, is a hardware unit that routes instructions to queues 266, 269, 272, and 275. In one  
25 embodiment, when a read instruction is encountered in the instruction stream, dispersal unit 260 examines the information in chain affinity unit 263 to determine whether a cluster has recently written to a register. If a cluster has recently written to a register that is to be read by the read instruction encountered in the instruction stream, then dispersal unit 260 routes the current read  
30 instruction to the queue that feeds the cluster. In that way, the current read instruction need not wait for retirement of the results of the previous write instruction before executing. This improves processor utilization. In another embodiment, dispersal unit 260, in addition to routing instructions, load balances

across the queues by maintaining a previous history of chains. In still another embodiment, dispersal unit 260, in addition to routing instructions, load balances across the queues by maintaining approximately equal queue lengths. In yet another embodiment, dispersal unit 260, in addition to routing instructions, load  
5 balances across the queues by maintaining a list of hints from the compiler.

Chain affinity unit 263 stores information. In one embodiment, the information to be stored in chain affinity unit 263 associates a cluster in the plurality of clusters with a register. The cluster associated with the register is the last cluster to write to the register. Any storage region is suitable for use as  
10 chain affinity unit 263. In one embodiment, the storage region is a table having a number of entries. In an alternate embodiment, the table has a number of entries equal to the number of registers in the architectural state of the system. Each entry consists of two fields. The first field is a one bit field that indicates whether the register was a destination of a write operation. The second field  
15 employs a sufficient number of bits to indicate the cluster to which the last write instruction was directed. For example, in an embodiment that includes four clusters and eight execution units, the number of bits in the second field is two.

In one embodiment, information is entered into chain affinity unit 263 by dispersal unit 260 each time a write instruction occurs in the instruction stream.  
20 Information identifying the register to be written, and information identifying the cluster to execute the write instruction is entered into chain affinity unit 263. In one embodiment, the physical location of information in chain affinity unit 263 can be used to identify the type of information stored. For example, information identifying the register to be written may be identified by the location in which  
25 the cluster information is entered into chain affinity unit 263. Information entered into chain affinity unit 263 is erased or removed by dispersal unit 260 at a time after write information is committed to the architectural state in retirement unit 218.

Intra-cycle dependency analyzer 278, in one embodiment of the present  
30 invention, is a hardware unit that analyzes groups of instructions in order to identify dependent instructions. A dependent instruction is an instruction that reads a register that was previously written. In one embodiment, a group of instructions is received from instruction fetch 203 and set for execution during

one clock cycle. Intra-cycle dependency analyzer 278 sorts the group of instructions received from instruction fetch 203 in order to identify instructions that may have dependencies, identifies dependent instructions in the sorted group, and communicates the identity of the dependent instructions to dispersal unit 260.

Queues, in one embodiment, buffer instructions for execution by the execution units. In one embodiment, queues 266, 269, 272, and 275 are first-in-first-out (FIFO) queues. A FIFO queue receives a series of inputs and disperses them in the order in which they were received. In an alternate embodiment, queues 266, 269, 272, and 275 are data flow schedulers. A data flow scheduler receives a series of inputs and is capable of dispersing the received inputs in an order other than the order in which they are received. Those skilled in the art will recognize that a queue suitable for use in the present invention can be realized in a variety of embodiments and is not limited to FIFO queues and data flow schedulers.

An advantage of an embodiment in which queues buffer instructions prior to their execution over a system without queues is that a less complicated instruction routing algorithm can be employed by dispersal unit 160. In a system without queues, dispersal unit 260 tracks the status of eight execution units in order to route the instructions. An algorithm employed by dispersal unit 260 to track the status of four queues can be less complicated than an algorithm employed by dispersal unit 260 to track of the status of eight execution units.

In one embodiment, queues 266, 269, 272, and 275 receive instructions from dispersal unit 260 and in turn route instructions to clusters 206, 209, 212, and 215. Dispersal unit 260, in one embodiment, manages chain affinity unit 263 by entering write information into chain affinity unit 263, and selectively erasing information from chain affinity unit 263. A time for entering write information into chain affinity unit 263 is when a write instruction is received. A time for selectively erasing information in chain affinity unit 263 is after a register, which was the destination of a write instruction, is committed to retirement unit 218.

One function of queues 266, 269, 272, and 275 is to stage a cache of instructions for execution at the input to each cluster 206, 209, 212, and 215,

respectively. Staging a cache of instructions at the input to each cluster allows high utilization of the instruction execution units, since the instruction execution units will have instructions to execute, as long as the queues do not completely drain.

- 5           Although specific embodiments have been illustrated and described herein, it will be appreciated by those of ordinary skill in the art that any arrangement which is calculated to achieve the same purpose may be substituted for the specific embodiment shown. This application is intended to cover any adaptations or variations of the present invention. Therefore, it is manifestly
- 10 intended that this invention be limited only by the claims and the equivalents thereof.



What is claimed is:

1. An apparatus for routing computer instructions comprising:  
a plurality of queues to buffer instructions to a plurality of clusters;  
5 a chain affinity unit to store information; and  
a dispersal unit to route instructions to the plurality of queues based on  
information to be stored in the chain affinity unit.
2. The apparatus of claim 1, wherein information to be stored in the chain  
10 affinity unit associates a cluster in the plurality of clusters with a register.
3. The apparatus of claim 2, wherein the dispersal unit is capable of routing  
an instruction to the cluster associated with the register.
- 15 4. The apparatus of claim 1, wherein each of the plurality of queues is a  
first-in-first-out (FIFO) queue.
5. The apparatus of claim 1, wherein each of the plurality of queues is data  
flow scheduler.
- 20 6. The apparatus of claim 1, wherein the chain affinity unit is a table.
7. The apparatus of claim 1, further comprising a retirement unit having a  
number of registers, and the retirement unit to receive input from the plurality of  
25 clusters.
8. The apparatus of claim 7, wherein the chain affinity unit is capable of  
having a number of entries equal to the number of registers in the retirement unit.
- 30 9. The apparatus of claim 8, wherein the dispersal unit routes instructions that  
read a register to one of the plurality of clusters that has most recently written the  
register.

10. The apparatus of claim 1, further comprising an intra-cycle dependency analyzer coupled to the dispersal unit, the intra-cycle dependency analyzer to identify dependencies in a group of instructions to the dispersal unit.

5 11. The apparatus of claim 10, wherein dependencies comprise a first instruction and a second instruction, where the second instruction is to read a register previously written by the first instruction.

12. An apparatus for routing computer instructions comprising:  
10 a plurality of queues to buffer instructions and to provide input to a plurality of clusters;  
a chain affinity unit having a write entry to indicate whether a register was a destination of a write operation, and a cluster entry to indicate one of the plurality of clusters to which the write operation was  
15 directed; and  
a dispersal unit that is coupled the chain affinity unit and the plurality of queues and that routes a read instruction to one of the plurality of clusters indicated by the cluster entry in the chain affinity unit, if the write entry in the chain affinity unit indicates that the register  
20 was the destination of a write operation.

13. The apparatus of claim 12, further comprising a retirement unit having a number of registers and the retirement unit to receive input from the plurality of clusters.

25

14. The apparatus of claim 13, wherein the chain affinity unit is capable of having a number of entries equal to the number of registers in the retirement unit.

15. The apparatus of claim 12, wherein the dispersal unit routes instructions to  
30 the plurality of clusters in a manner that maintains approximately equal queue lengths.

16. The apparatus of claim 12, further comprising an intra-cycle dependency analyzer coupled to the dispersal unit, and where the intra-cycle dependency analyzer is to identify dependencies in a group of instructions to the dispersal unit.

5 17. A method for routing a computer instruction comprising:  
determining if an instruction reads a register; and  
routing the instruction, if it reads the register, to a cluster in response to  
information in a chain affinity unit.

10 18. The method of claim 17 further comprising:  
buffering the instruction in a queue.

19. The method of claim 17 further comprising:  
selectively erasing information in the chain affinity unit.

15

20. The method of claim 18, further comprising:  
selectively erasing information in the chain affinity unit.

21. A method for routing a computer instruction comprising:  
20 determining if an instruction has an intra-cycle dependency; and  
routing the instruction to a cluster in response to the dependency.

22. The method of claim 21, wherein determining if an instruction has an  
intra-cycle dependency comprises identifying a first instruction and a second  
25 instruction to be executed in one cycle, where the second instruction is to read a  
register previously written by the first instruction.

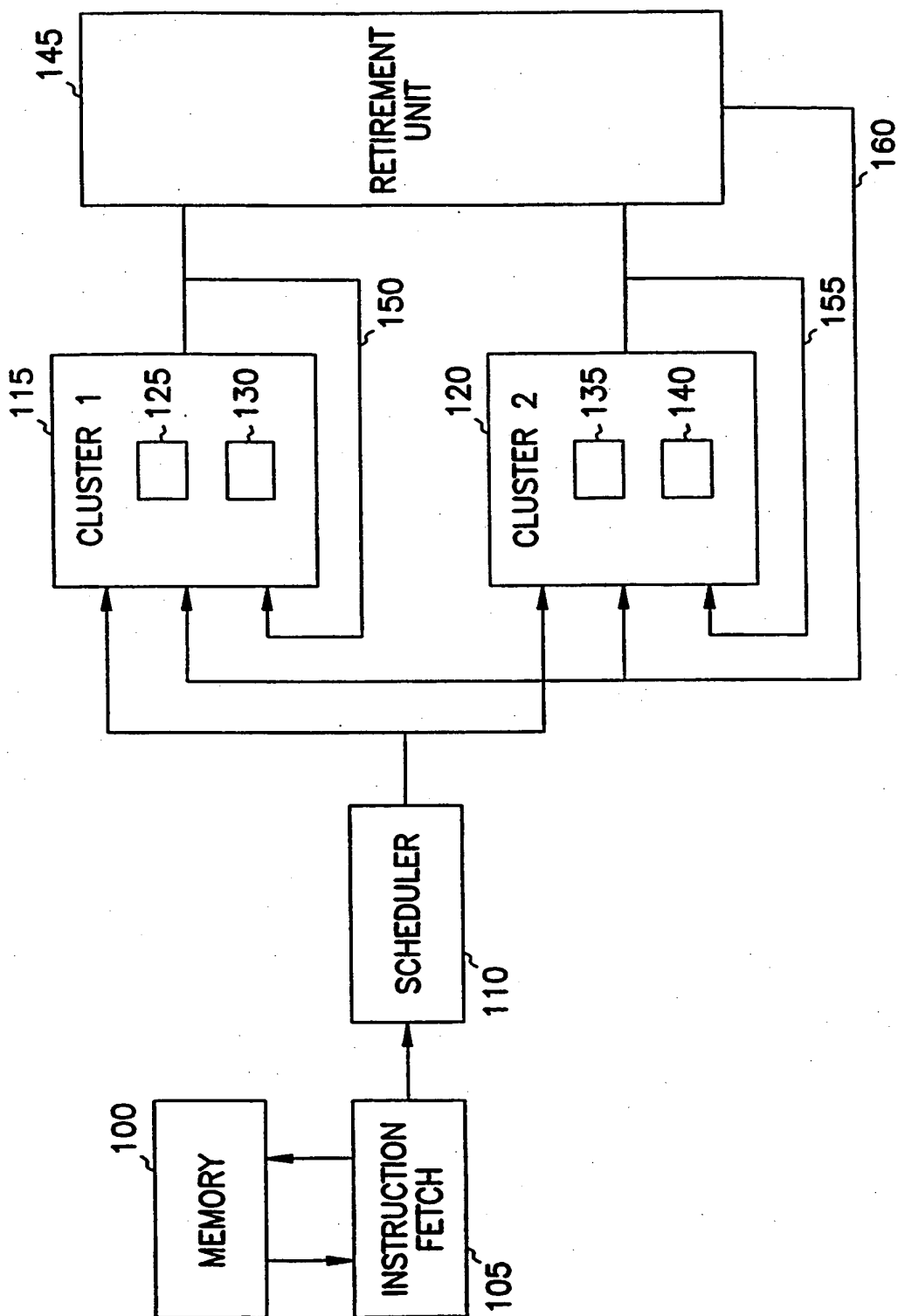


FIG. 1  
(PRIOR ART)

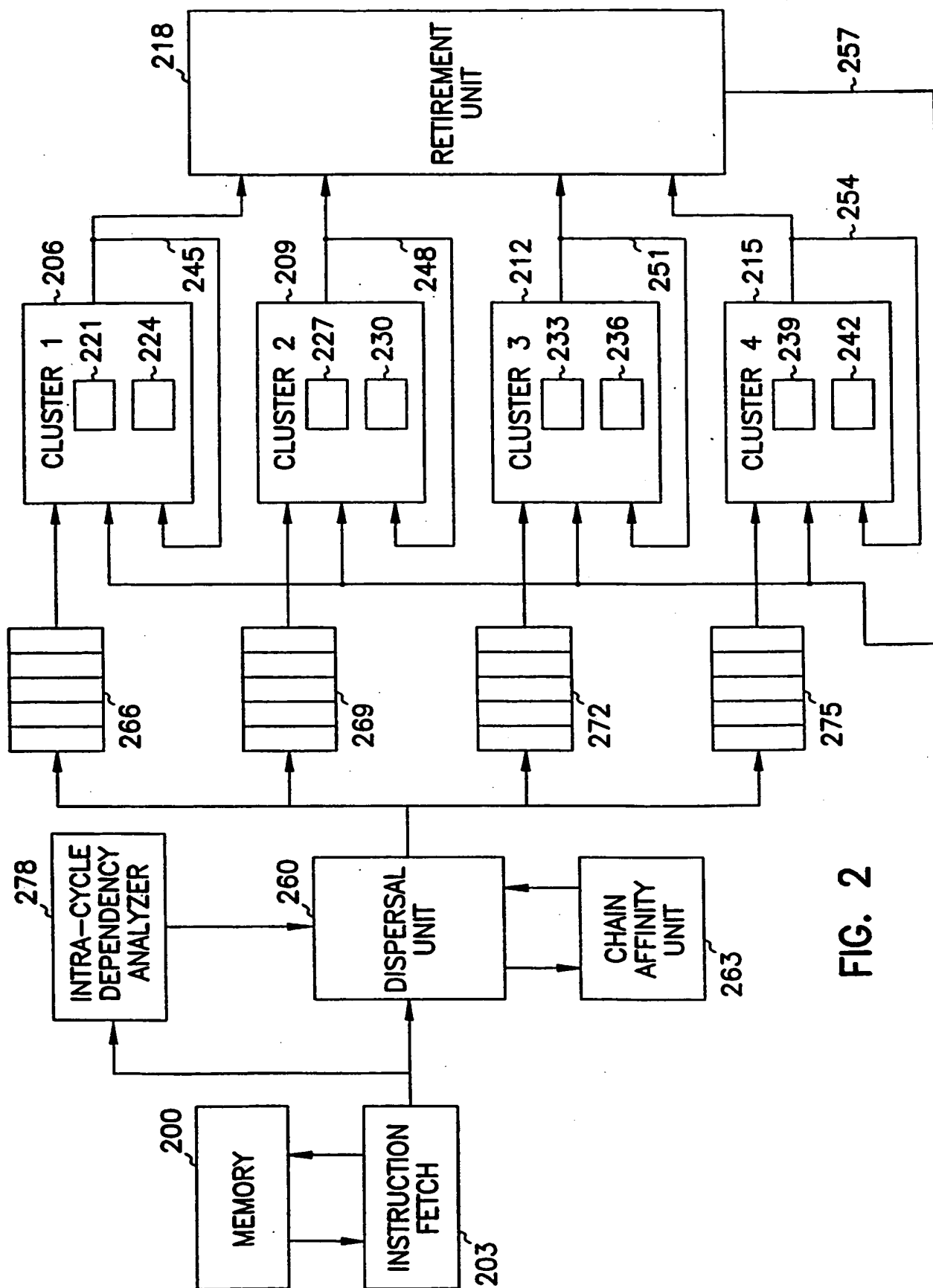


FIG. 2

# INTERNATIONAL SEARCH REPORT

International Application No

PCT/US 99/30480

A. CLASSIFICATION OF SUBJECT MATTER  
IPC 7 G06F9/38

According to International Patent Classification (IPC) or to both national classification and IPC

## B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

IPC 7 G06F

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)

## C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category *	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	PALACHARLA S ET AL: "COMPLEXITY-EFFECTIVE SUPERSCALAR PROCESSORS" ANNUAL INTERNATIONAL SYMPOSIUM ON COMPUTER ARCHITECTURE, US, NEW YORK, ACM, vol. CONF. 24, 1997, pages 206-218, XP000738158 ISBN: 0-7803-4175-9 pages 213-218, section 5	1-4, 6-18, 21, 22
X	GB 2 322 718 A (IBM) 2 September 1998 (1998-09-02) the whole document	1-4, 17, 21, 12
A		
X	WO 95 09394 A (APPLE COMPUTER) 6 April 1995 (1995-04-06) the whole document	1, 10, 11, 17, 18
	---	
	---/---	

☒ Further documents are listed in the continuation of box C.

☒ Patent family members are listed in annex.

### \* Special categories of cited documents:

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

Date of the actual completion of the international search

28 April 2000

Date of mailing of the international search report

08/05/2000

Name and mailing address of the ISA

European Patent Office, P.B. 5818 Patentlaan 2  
NL - 2280 HV Rijswijk  
Tel. (+31-70) 340-2040, Tx. 31 651 epo nl,  
Fax: (+31-70) 340-3016

Authorized officer

Klocke, L

# INTERNATIONAL SEARCH REPORT

International Application No

PCT/US 99/30480

## C.(Continuation) DOCUMENTS CONSIDERED TO BE RELEVANT

Category *	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	EP 0 767 425 A (DIGITAL EQUIPMENT CORP) 9 April 1997 (1997-04-09) the whole document ----	1-22
A	US 5 699 537 A (SHARANGPANI ET AL) 16 December 1997 (1997-12-16) the whole document -----	1,12,17, 21

# INTERNATIONAL SEARCH REPORT

Information on patent family members

International Application No

PCT/US 99/30480

Patent document cited in search report		Publication date	Patent family member(s)		Publication date
GB 2322718	A	02-09-1998	US 5864341	A	26-01-1999
WO 9509394	A	06-04-1995	US 5481743	A	02-01-1996
			AU 7841794	A	18-04-1995
			US 5826096	A	20-10-1998
EP 0767425	A	09-04-1997	NONE		
US 5699537	A	16-12-1997	NONE		